Lab Exercise
CPE 357
Systems Programming

CAL POLY
SAN LUIS OBISPO

LAB 1

Exercise:

Write a double linked list with strings (char arrays) as data.

Howto:

```
//Use this structure:
struct listelement
{
listelement *next,*prev;
char text[1000];
};
//and this global list head:
listelement *head = NULL;
```

You can also use “class” instead of “struct” and do it OOP in C++, but that’s up to you.

You program should start with printing a menu with following items:
Select:
1 push string
2 print list
3 delete item
4 end program

Scanf a number from the keyboard and execute the selected item.

push string
Reads a string, generates a new list item and the end of the list and assigns the elements data with the string. Program should go back to the start menu.

print list
prints the list on the screen. Each item one row. Program should go back to the start menu.

delete item
Reads a number from the keyboard corresponding to a list-item. The first item is “1”. It deleted this item. Should th eitem not exist… I leave it up to you how to handle that. Program should go back to the start menu.

end program
Deletes the whole list and ends the program.

<u>Submission:</u>

On Canvas.

<u>Any style must do's?</u>

Use function or not, (I would strongly recommend that!) that's up to you, but we will check if you did use a double linked list and also if you linked it forth and back! You have any ideas to make it nicer (additional text, whatever): Feel free to do so, but don't deviate from the core idea. Other than that, please only one file! This will not be graded by an automatic submission system, but by hand. Use your creativity. Don't cheat, we will run MOSS tests.

<u>Hints:</u>

A function which returns the last element of the list would be a great idea!
Then, if you push back a new item, generate this new item and link it to the last element, and the last element to it. Don't forget to set loose ends to "NULL".

Result example:

```
1 push string
2 print list
3 delete item
4 end program
>1
insert text [<-not a must do but looks nice]
>hello
done push string [<-not a must do but looks nice]

1 push string
2 print list
3 delete item
4 end program
>1
insert text [<-not a must do but looks nice]
>world
done push string [<-not a must do but looks nice]

1 push string
2 print list
3 delete item
4 end program
>2
hello
world

1 push string
2 print list
3 delete item
4 end program
>4
```